Time-Independent Discrete Gaussian Sampling for Post-Quantum Cryptography

Time-Independent Discrete Gaussian Sampling For Post-Quantum Cryptography

A. Khalid, J. Howe, C. Rafferty, and M. O'Neill

Centre for Secure Information Technologies (CSIT),
Queen's University Belfast, UK.

***Abstract*—As the development of a viable quantum computer nears, existing widely used public-key cryptosystems, such as RSA, will no longer be secure. Thus, significant effort is being invested into post-quantum cryptography (PQC). Lattice-based cryptography (LBC) is one such promising area of PQC, which offers versatile, efficient, and high performance security services. However, the vulnerabilities of these implementations against side-channel attacks (SCA) remain significantly understudied. Most, if not all, lattice-based cryptosystems require noise samples generated from a discrete Gaussian distribution, and a successful timing analysis attack can render the whole cryptosystem broken, making the discrete Gaussian sampler the most vulnerable module to SCA. This research proposes countermeasures against timing information leakage with FPGA-based designs of the CDT-based discrete Gaussian samplers with constant response time, targeting encryption and signature scheme parameters. The proposed designs are compared against the state-of-the-art and are shown to significantly outperform existing implementations. For encryption, the proposed sampler is 9x faster in comparison to the only other existing time-independent CDT sampler design. For signatures, the first time-independent CDT sampler in hardware is proposed.**

I. Introduction

Cryptography is one of the most important tools to protect information sent across public networks, using digital signatures and encryption. This security, currently supported by the hardness of factoring large primes (RSA) and the discrete logarithm problem (elliptic-curve cryptography), may soon be under threat by the possible construction of quantum computers. Indeed, the NSA and CESG have both indicated a need to transition towards quantum-resistant algorithms [1], [2]. Protecting secure communications against quantum attacks is vital, and thus several post-quantum (or quantum-resilient) constructions have been proposed to protect technologies such as cloud security and the Internet of things. Lattice-based cryptography (LBC) is arguably the most promising when compared to other post-quantum cryptosystems, as it offers extended functionality and average-case to worst-case hardness, whilst being more efficient for both encryption and digital signature schemes [3].

However, the real-world practicality of LBC should be considered, including suitable countermeasures against side-channel analysis (SCA). A NIST call [4] requests new quantum-resilient algorithms that offer SCA attack resistance. The most vulnerable component of lattice-based cryptosystems is the generation of randomness, typically discrete Gaussian randomness, to mask the computations of the secret-key and plaintext data. Unfortunately, discrete Gaussian samplers are highly susceptible to timing-analysis attacks, due to non-constant run-time [5]. There has been little research into the SCA-resilience of lattice-based cryptographic implementations to physical attacks; only Roy *et al.* have investigated masking [6] and side-channel secure discrete Gaussian sampling [7].

This research proposes a timing-attack resilient hardware design of a discrete Gaussian sampler, adopting the cumulative distribution table (CDT) [8] technique. Practical FPGA designs of novel CDT-based constant response time samplers, with appropriate practical parameters for both encryption and signatures, are presented.

II. Background

A. Lattice-based Cryptography

One foundational work that underpins LBC is the learning with errors (LWE) problem [9]. Cryptosystems based upon the LWE problem enjoy worst-case to average-case hardness; they are proven infeasible to break unless all instances of certain lattice problems are easy to solve [10], [11]. Lattice-based cryptosystems, based on the hardness of the LWE problem are as hard to solve as Definition 1.

Definition 1: [LWE] For positive integers n and $q \geq 2$, the secret $\mathbf{s} \in \mathbb{Z}_q^n$, and a probability distribution χ on $\mathbb{Z}_q$, let $A_{\mathbf{s},\chi}$ be the LWE distribution, obtained by choosing $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$, a noise term $e \leftarrow \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$. The decisional LWE problem is defined as, given access to m independent samples chosen according to $A_{\mathbf{s},\chi}$, distinguish between these m LWE samples and ones chosen uniformly at random with noticeable probability.

B. Discrete Gaussian Sampling

The error distribution χ seen in Definition 1 is almost always defined as the discrete Gaussian distribution[†]. The centered discrete Gaussian distribution $D_{\mathbb{Z},\sigma}$, over $\mathbb{Z}$, with standard deviation σ, is defined proportionally such that a value $x \in \mathbb{Z}$ is sampled from D_σ with the probability $\rho_\sigma(x)/\rho_\sigma(\mathbb{Z})$, where:

$$\rho_\sigma(x) = \exp\left(\frac{-x^2}{2\sigma^2}\right) \text{ and } \rho_\sigma(\mathbb{Z}) = \sum_{k=-\infty}^{\infty} \rho_\sigma(k).$$

[†]With the exception of key-exchange protocols such as [12], which are able to employ the slightly less "normal", binomial distribution.

Considering $S_\sigma = \rho_\sigma(\mathbb{Z}) = \sum_{k=-\infty}^{\infty} \rho_\sigma(k) \approx \sqrt{2\pi}\sigma$, the probability of sampling $x \in \mathbb{Z}$ from the distribution $D_{\mathbb{Z},\sigma}$ is calculated as $\rho_\sigma(x)/S_\sigma$.

1) Exploiting Symmetry: To half the initial memory requirement, one can consider the distribution over $\mathbb{Z}^+$, proportional to $\rho_\sigma(x)$ for $\forall x > 0$. For $x = 0$, $\rho_\sigma(0)$ is halved, otherwise this will be counted twice. The distribution can be recovered by adding a random sign bit after sampling.

2) Practical Discrete Gaussian Parameters: The statistical distance between the "perfect" theoretical discrete Gaussian distribution and the "practical" should be no greater than $2^{-\lambda}$. It is recommended [13] that precision need be no greater than $\lambda/2$, for a target security level λ-bits, as it is argued that no algorithm can distinguish between a "perfect" sampler and one with statistical distance $2^{-\lambda/2}$. Two important cryptographic applications for the samplers are targeted: encryption and digital signature schemes. The parameters are from the encryption scheme by Lindner and Peikert [14] (LP)[‡], where $(\sigma, \lambda, \tau) = (3.33, 64, 9.42)$, and the digital signature scheme by Ducas *et al.* [15] (BLISS), where $(\sigma, \lambda, \tau) = (215, 64, 9.42)$.

3) Gaussian Convolution: The standard deviation can be significantly decreased by using Peikert's convolution lemma [18], adapted by Pöppelmann *et al.* [19]. Referring to [8], [19] for the formal definitions of the *smoothing parameter* η and Kullback-Leibler divergence respectively, the adaption states:

Lemma 1: Let $x_1 \leftarrow D_{\mathbb{Z},\sigma_1}$, $x_2 \leftarrow D_{k\mathbb{Z},\sigma_2}$ for some positive real σ_1, σ_2 and let $\sigma_3^{-2} = \sigma_1^{-2} + \sigma_2^{-2}$ and $\sigma^2 = \sigma_1^2 + \sigma_2^2$. For any $\epsilon \in (0, \frac{1}{2})$ if $\sigma_1 \geq \eta_\epsilon(\mathbb{Z})/\sqrt{2\pi}$ and $\sigma_3 \geq \eta_\epsilon(k\mathbb{Z})/\sqrt{2\pi}$, then ("perfect") distribution $\mathcal{P}$ of $x_1 + x_2$ verifies

$$D_{KL}(\mathcal{P}||D_{\mathbb{Z},\sigma}) \leq 2\Big(1 - \Big(\frac{1+\epsilon}{1-\epsilon}\Big)^2\Big)^2 \approx 32\epsilon^2.$$

Proof: The proof of this lemma is referred to in [19]. ■

Lemma 1 holds for $\sigma_{\text{BLISS}} = 215$ by setting $k = 11$, such that $\sigma' = \sigma/\sqrt{1+k^2} \approx 19.47$, and by sampling twice $x'_1, x'_2 \leftarrow D_{\mathbb{Z},\sigma'}$ a value $x \leftarrow D_{\mathbb{Z},\sigma}$ can be built as $x = x'_1 + kx'_1$. Although an additional sample is required, a smaller standard deviation means that memory consumption of the precomputed tables is significantly reduced; memory consumption is reduced from 130kb for $\sigma_{\text{BLISS}} = 215$, to 11.74kb for $\sigma'_{\text{BLISS}} = 19.47$.

III. The Cumulative Distribution Table Sampler

The cumulative distribution table (CDT) sampler requires a precomputed table of discrete Gaussian cumulative distribution function (CDF) values [8]. CDT sampling is arguably more promising than other discrete Gaussian sampling schemes, as the distribution parameters are known in advance. The CDF values range from $0 \leq x \leq 1$, and are stored in a look-up table $S[x]$, where the total number of table entries is at least: $N = \tau \times \sigma$, where $0 = S[0] < S[1] < \ldots < S[N-1] = 1$. CDT sampling works as follows: sample $r \xleftarrow{\$} \{-\tau\sigma, \ldots, \tau\sigma\}$, with λ bits of precision. r is compared against the CDF table contents to find an interval such that: $S[x] \leq r < S[x+1]$, where x is output as the required discrete Gaussian sample, occurring with probability $\rho[x] = S[x+1] - S[x]$.

For comparisons, binary search is chosen, and is detailed in Algorithm 1. Pointers *min*, *max*, and *mid* point to the first, last, and middle of the search space, respectively. r is iteratively compared to the middle value of the search space $S[cur]$, whose upper or lower half is discarded depending on the comparison result. For a finite search space with N samples, the comparisons required before a match does not exceed $\lceil \log_2(N) \rceil$.

Algorithm 1 CDT Sampling from $D_{\mathbb{Z},\sigma}$ via Binary Search

Require:
Three integers min, mid, and max.
CDT values $0 = S[0] < S[1] < \cdots < S[N-3] = 1$.
Uniformly sampled $r \in \{0, \ldots, (2^\lambda - 1)\}$ and bit $b \in \{0, 1\}$.
Ensure: $min \leftarrow 0; max \leftarrow N; mid \leftarrow (min + max)/2;$
while $(max > min)$ **do**
  if $(r \geq S[mid])$ **then**
    $min \leftarrow (mid + 1);$
  else
    $max \leftarrow mid;$
return $x = (-1)^b(mid - 1)$

A. Previous work

The use of CDT sampling was proposed by Peikert [8], and adapted by Ducas *et al.* [15]. Pöppelmann *et al.* [19] implemented BLISS on reconfigurable hardware and suggested optimisations for the CDT sampler, including hashing to reduce the search space and skipping the leading zero storage to reduce the table size by a factor of 2. Du and Bai [20] further optimised hardware area by using piecewise comparison and hashing. These optimisations reduce the precomputed table size, and improve throughput. However, hashing divides the search intervals into irregular sizes, meaning the binary search has non-constant execution time, making it susceptible to timing analysis attacks.

The only CDT sampler design on a FPGA with constant-time throughput is by Pöppelmann and Güneysu [21]. This fully pipelined design offers a single cycle per sample throughput, but the large number of parallel comparisons renders it impractical. Roy *et al.* [7] presented a hardware design of a discrete Gaussian sampler resistant against timing attacks, using a Knuth-Yao sampler that generates a batch of samples, subsequently shuffled to disassociate the related timing information. However, this design is non-constant time and only suitable for small standard deviations.

B. Timing Attack Vulnerabilities

Side channel attacks are physical attacks, based on information gained from the physical implementation of a cryptosystem. To date, little research has been conducted on the vulnerabilities of LBC implementations to physical attacks; efforts so far are summarised by Hodgers *et al.* [22]. Timing analysis attacks are highly algorithm-specific in nature, where the dependency between the execution time of an algorithm and its secret internal states is exploited. Attacks on LBC constructions are emerging [23], [5]. The timing-attack countermeasure is to guarantee an execution time independent of the secret values [24]. This can be achieved by ensuring

[‡]The same parameters are used for implementing the ring-LWE encryption scheme of Lyubashevsky *et al.* [16], see [17] for a hardware implementation.

constant response time [21] or subsequent random shuffling of the secret values [7]. The following definition of time independence is used in this research:

Definition 2 (Time independence): A program achieves the property of *independent-time* when no information about the secret value(s) is leaked by the timing of the program.

IV. Constant-Time CDT Hardware Architectures

A constant-time implementation of a CDT sampler is achievable by comparing the table in a fixed number of clock cycles. Algorithm 1 shows that an early termination is possible if the comparison of uniformly sampled r and the $S[mid]$ returns an equality. This exact match could happen with a small probability of $2^{-\lambda}N$. This early termination is avoided by not monitoring an exact match of r and $S[mid]$ separately. Hence, the binary search algorithm is always bounded between $\left[\lfloor \log_2(N) \rfloor, \lceil \log_2(N) \rceil\right]$ search iterations of the for loop. Consequently, where N is a power of two, the algorithm executes exactly in constant-time; and for all other N, the algorithm is tweaked to occasionally perform an extra iteration to ensure the algorithm complexity is fixed to $\lceil \log_2(N) \rceil$.

A. Trivium as a PRNG

The CDT sampler requires uniformly distributed samples, for which Trivium [25] is selected, due to its versatility. It is a synchronous, binary stream cipher with a 288-bit internal state. To achieve a large number of uniformly random bits per clock cycle, the Trivium modules are unrolled. The resources for unrolled designs compared to standard Trivium×1 are rather negligible: 28 additional LUTs, 63 additional flip-flops, and 15 additional slices for Trivium×8 and 26 additional LUTs, 147 additional flip-flops, and 21 additional slices for Trivium×32 on a Spartan-6 LX25-3 device, post place-and-route.

B. Proposed Constant-Time CDT Sampler For Encryption

Figure 1a illustrates the proposed constant-time CDT sampler for encryption. The CDF table $S[\cdot]$ consists of $N = 32$ $(\tau \times \sigma)$ entries, with $\lambda = 64$. A single ported ROM, a 5-bit address port and a 64-bit data port are employed. Trivium×64 is used for uniform sample generation, with module initialisation (key setup, IV setup, and the randomisation phase) handled externally at startup, and thereafter controlled by the binary search state-machine, `BinSearch`. The uniform samples are only generated when required, saving circuit power.

The `BinSearch` state-machine begins at the `SET` state, resetting the three pointers (`min`, `mid`, and `max`) to initial values (as in Algorithm 1). It transitions unconditionally to the `SEARCH` state in the next clock cycle and these three pointers are updated, given the result from the 64-bit comparison. After exactly 5 cycles, a search is found, and the state generates a single bit `hit`. The Trivium module is activated by this `hit` to request a new uniformly random 64-bit value. The buffered `mid` is combined with a random bit b, which attaches a sign to the generated discrete Gaussian sample `x`.

C. Proposed Constant-Time CDT Sampler For Signatures

The proposed CDT sampler for signatures uses two state-machines, `BinSearch0` and `BinSearch1`, to parallelise two independent searches (see Figure 1b). Since most FPGA devices have dual ported BRAMs, the CDF table for both state-machines can be accessed from one BRAM. Each state-machine has an 8-bit address and a 64-bit data port. The two state machines each get a 64-bit uniformly random number, `r0` and `r1` respectively, from the PRNG, in two consecutive clock cycles. During the next 8 clock cycles, `r0` and `S[mid0]` are compared in `BinSearch0` and the three pointers are updated. Simultaneously, `r1` is processed at `BinSearch1`. The state-machines work independently to generate two independent random samples x_0 and x_1 in 8 clock cycles; the two samples are then combined as $x = x_0 + 11x_1$, where lastly a sign bit is assigned.

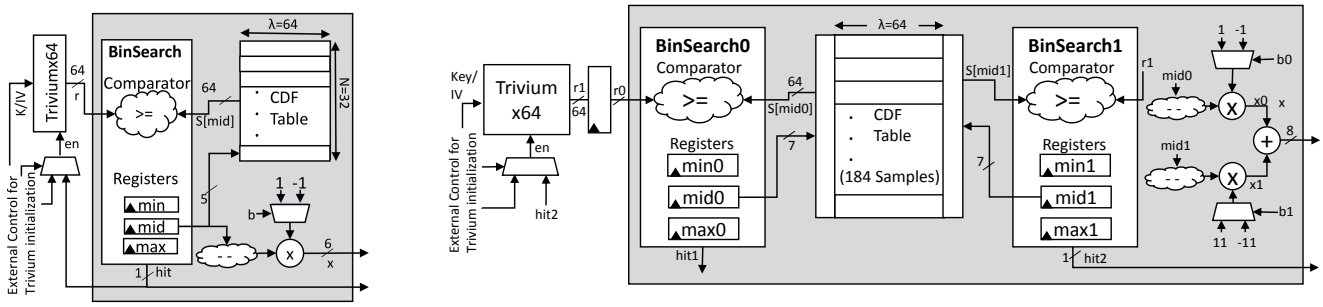
V. Results, Evaluation and Comparison

This section presents the post place-and-route performance results and comparison of the proposed samplers with existing sampler implementations for encryption and signatures. Xilinx ISE 14.7 is used, and where possible comparable implementations have been re-run on the same FPGA device in order to fairly compare the results. Throughput and throughput per area (TPAR) have been evaluated for all schemes, in terms of sampling operations per second (Ops/s) and sampling operations per second per slice (Ops/s/S). Table I gives the CDT sampler resource consumption for both encryption and signature parameters. A low cost Spartan-6 FPGA is targeted, and low area and balanced results are presented, where the area-optimised designs employ BRAM, unlike the balanced designs, which offer higher running frequencies.

Table II shows performance results of the samplers, compared with existing CDT hardware samplers. For encryption parameters, a single ported distributed ROM comprising of LUTs is proposed in the design without BRAM. The slice resources can be significantly reduced if BRAMs are utilised. However, the price of resource reduction is paid for by reduced operable frequency. The only other constant-time CDT implementation for encryption is by Pöppelmann and Güneysu [21], which generates a single sample per cycle. However, it is 4x slower in frequency with 5x many slices, and thus this research offers a more lightweight, constant-time alternative. The CDT sampler design by Du and Bai [20] is lightweight but it is only for encryption and does not run in independent-time. For signature parameters, the implementation by Pöppelmann *et al.* [19] operates in non-constant time but has a lower throughput per slice than this work. Thus the CDT sampler proposed in this research is preferable for practical implementations.

Table I: Resource consumption of the CDT sampler on a Spartan-6 LX25-3 FPGA for a) encryption parameters and b) signature scheme parameters.

Sampler	Registers	LUT	Slice	BRAM	Freq.
a) CDT_Area	17	53	16	2	136
a) CDT_Balance	26	66	21	0	394
b) CDT_Area	48	130	44	2	126
b) CDT_Balance	64	577	179	0	130
Available	30,064	24,051	3,758	52	-



(a) CDT sampler for encryption parameters.

(b) CDT sampler for digital signature scheme parameters.

Figure 1: The CDT-based discrete Gaussian samplers for encryption and digital signature scheme parameters, using one `BinSearch` state machine for (a) and two `BinSearch` state machines for (b), each accessing the CDF table.

Table II: Post-place and route results of the proposed constant-time CDT sampler for encryption and signature parameters, in comparison to existing CDT-based results.

Op.	Implementation	Device	λ	LUT/FF /Slice	BRAM	Freq. (MHz)	Clock Cycles	Rand. Bits	Ops/s (MHz)	Ops/s/S (MHz/S)	Time Ind.
$\sigma_{\text{LP}} = 3.33$	Pöppelmann & Güneysu [21]	6VLX75T-2	80	863/6/231	0	61	1	85	6.1	0.26	✓
				911/6/255	0	61	1	85	6.1	0.24	✓
	This work	6VLX75T-2	64	112/19/43	0	297	5	64	59.4	1.38	✓
				53/17/15	1	193	5	64	38.6	2.57	✓
	Du & Bai [20]	5VLX30	112	43/33/17	1	259	≈2.28	≈9.44	113.6	6.68	✗
				85/65/39	1	256	≈1.14	≈9.44	224.6	5.76	✗
$\sigma_{\text{BLISS}} = 215$	Pöppelmann *et al.* [19]	6SLX25-3	128	928/1121/299	1	129	≈7.5	≈21	17.2	0.06	✗
	This work	6SLX25-3	64	577/64/179	0	130	8	64	16.3	0.09	✓
				130/48/44	2	126	8	64	15.8	0.36	✓

VI. Conclusion

In this research two independent-time hardware designs of a discrete Gaussian CDT sampler are proposed, suitable for encryption and signature applications, with a focus on low-area foot-print and high throughput. Resistance against timing attacks is achieved by ensuring constant execution time. Moreover, the proposed hardware CDT sampler designs clearly outperform the previously proposed samplers.

References

[1] CNSS, "Use of public standards for the secure sharing of information among national security systems," Committee on National Security Systems: CNSS Advisory Memorandum, Information Assurance 02-15, July 2015.

[2] CESG, "Quantum key distribution: A CESG white paper," February 2016. [Online]. Available: https://www.cesg.gov.uk/white-papers/quantum-key-distribution

[3] J. Howe, T. Pöppelmann, M. O'Neill, E. O'Sullivan, and T. Güneysu, "Practical lattice-based digital signature schemes," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 14, no. 3, p. 41, 2015.

[4] D. Moody, "Post-quantum cryptography: NIST's plan for the future," Talk given at PQCrypto '16 Conference, 23-26 February 2016, Fukuoka, Japan, February 2016. [Online]. Available: https://pqcrypto2016.jp/data/pqc2016_nist_announcement.pdf

[5] L. G. Bruinderink, A. Hülsing, T. Lange, and Y. Yarom, "Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme," in *CHES*, 2016, pp. 323–345.

[6] O. Reparaz, S. S. Roy, F. Vercauteren, and I. Verbauwhede, "A masked ring-LWE implementation," in *CHES*, 2015, pp. 683–702.

[7] S. S. Roy, O. Reparaz, F. Vercauteren, and I. Verbauwhede, "Compact and side channel secure discrete Gaussian sampling," *IACR Cryptology ePrint Archive*, vol. 2014, p. 591, 2014.

[8] C. Peikert, "An efficient and parallel Gaussian sampler for lattices," in *CRYPTO*, 2010, pp. 80–97.

[9] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *STOC*, 2005, pp. 84–93.

[10] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," in *STOC*, 1996, pp. 99–108.

[11] O. Regev, "The learning with errors problem," *Invited survey in CCC*, 2010.

[12] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key exchange - A new hope," in *USENIX*, 2016, pp. 327–343.

[13] M.-J. O. Saarinen, "Gaussian sampling precision and information leakage in lattice cryptography," Cryptology ePrint Archive, Report 2015/953, 2015.

[14] R. Lindner and C. Peikert, "Better key sizes (and attacks) for LWE-based encryption," in *CT-RSA*, 2011, pp. 319–339.

[15] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky, "Lattice signatures and bimodal Gaussians," in *CRYPTO (1)*, 2013, pp. 40–56, full version: https://eprint.iacr.org/2013/383.pdf.

[16] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *EUROCRYPT*, 2010, pp. 1–23.

[17] T. Pöppelmann and T. Güneysu, "Area optimization of lightweight lattice-based encryption on reconfigurable hardware," in *ISCAS*, 2014, pp. 2796–2799.

[18] D. Micciancio and C. Peikert, "Hardness of SIS and LWE with small parameters," in *CRYPTO (1)*, 2013, pp. 21–39.

[19] T. Pöppelmann, L. Ducas, and T. Güneysu, "Enhanced lattice-based signatures on reconfigurable hardware," in *CHES*, 2014, pp. 353–370, full version: https://eprint.iacr.org/2014/254.pdf.

[20] C. Du and G. Bai, "Towards efficient discrete Gaussian sampling for lattice-based cryptography," in *FPL*. IEEE, 2015, pp. 1–6.

[21] T. Pöppelmann and T. Güneysu, "Towards practical lattice-based public-key encryption on reconfigurable hardware," in *SAC*, 2013, pp. 68–85.

[22] P. Hodgers, F. Regazzoni, R. Gilmore, C. Moore, and T. Oder, "State-of-the-art in physical side-channel attacks and resistant technologies." Technical report, 2016.

[23] J. H. Silverman and W. Whyte, "Timing attacks on NTRUEncrypt via variation in the number of hash calls," in *CT-RSA*. Springer, 2007, pp. 208–224.

[24] E. Tromer, D. A. Osvik, and A. Shamir, "Efficient cache attacks on AES, and countermeasures," *Journal of Cryptology*, vol. 23, no. 1, pp. 37–71, 2010.

[25] C. De Canniere and B. Preneel, "Trivium specifications. eSTREAM," *ECRYPT Stream Cipher Project, Report*, vol. 30, 2005.